如何把网页制作成个性化电子书 | 实用技巧

SpencerWoo 05月06日

编注（@Minja）：写文章时常常需要援引，需要回溯。对于各种存档、剪藏工具的不满，让我萌发了自己制作电子书的想法。正好 @Spencerwoo 在这方面有着扎实的折腾能力，便拜托他一起研究了一套简单易行的方法，并写成文章和诸位分享。

网络世界上有价值的文章虽然多，但不是每篇的阅读体验都尽如人意。如果想有一个优秀的文章阅读体验，至少要保证我们在阅读的文章：

- 起码，没有碍眼的牛皮癣广告
- 通常，排版统一，图片规范
- 最好，方便对文中关键词进行检索

很多时候，我们都依赖于浏览器的阅读模式，或是 Pocket 和 RSS 客户端进行阅读。但是阅读模式没法后期整理，多数阅读服务的全文检索功能又需要付费，而且网上的文章说不定哪天就找不着了。或许，将文章以**电子书**的形式保存到本地，是一个更加便于回溯的选择。

若是我们在网上看到一部以网页形式存在的电子书，想在本地直接保存整个网页（俗称「剪藏」），那简直是大费周

折。现有的网页剪藏工具不仅可能夹带上无用的、影响体验的广告和其他冗余区域，还可能丢失重要宝贵的内容。不仅如此，能够方便抓取图片、自定义保存至本地的文章样式的工具，更是几乎没有。所以，接下来给大家介绍一套免费制作**个性化电子书**的方法。

本文主要利用 **Pandoc** 这个开源工具，对于需要 MOBI 或 PDF 格式电子书的读者，文章后半部分也有简便的转换方法。

以 ePub 电子书为突破点

目前主流的电子书格式很多，但是本文主要推荐 ePub，它相对开放通用，和可以轻松转换成其他格式。

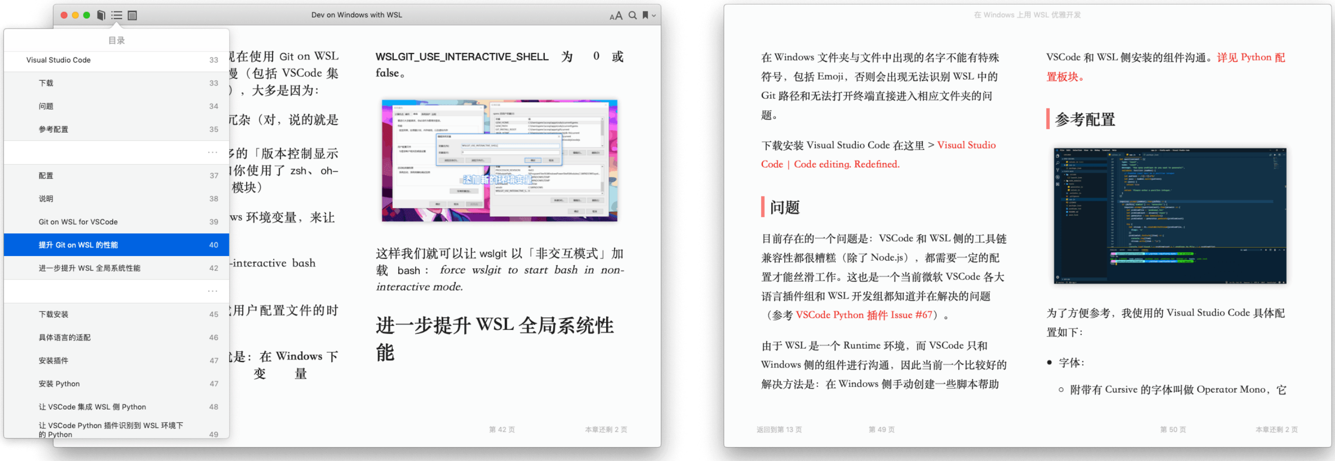
PDF
EPUB
MOBI
主流电子书的格式

主流电子书格式

大家最为熟知的电子书格式或许是纯文本 TXT 格式，但是 TXT 之所以被称作「纯文本」，就是因为他不支持分章节、不支持图片、不支持封面、不支持超链接。为了让电子书**有格式、有图片、有内容，**当前常见的电子书通常都是 PDF、ePub 和 MOBI 这三种格式的。其中：

- **PDF**：就是我们常见的刊物、文档的排版格式，能够最大程度的保证各种设备上文档样式的一致性，但是对小屏幕设备支持较差，很多时候都需要拖动放大才能看清内容。
- **ePub**：一种基于 HTML 的电子书格式，其排版效果接近 PDF，但是对小屏设备支持的更好，可以适应屏幕尺寸自动排版，体积往往也比 PDF 小。
- **MOBI**：是另外一种电子书的格式，也是 Kindle 上面的主流电子书格式。ePub 和 MOBI 这两个格式的文件往往可以无痛转换。

我们本篇文章的主要目的就是利用接下来要介绍的工具，自己制作一本**内嵌图片**、**包含目录**、**清晰精美**的电子书。相对而言，ePub 灵活性较好，目录、自定义排版一应俱全，也方便转换成其他两种格式，本文就从它切入。


Dev on Windows with WSL

目录

Visual Studio Code 33
下载 33
问题 34
参考配置 35
…
配置 37
说明 38
Git on WSL for VSCode 39
提升 Git on WSL 的性能 40
进一步提升 WSL 全局系统性能 42
…
下载安装 45
具体语言的适配 46
安装插件 47
安装 Python 47
让 VSCode 集成 WSL 侧 Python 48
让 VSCode Python 插件识别到 WSL 环境下的 Python 49

见在使用 Git on WSL
曼（包括 VSCode 集
），大多是因为：
冗杂（对，说的就是
乡的「版本控制显示
1你使用了 zsh、oh-
模块）
ws 环境变量，来让
-interactive bash
用户配置文件的时
是：在 Windows 下
变 量

WSLGIT_USE_INTERACTIVE_SHELL 为 0 或 false。

这样我们就可以让 wslgit 以「非交互模式」加载 bash：*force wslgit to start bash in non-interactive mode.*

进一步提升 WSL 全局系统性能

第 42 页 本章还剩 2 页

在 Windows 上用 WSL 优雅开发

在 Windows 文件夹与文件中出现的名字不能有特殊符号，包括 Emoji，否则会出现无法识别 WSL 中的 Git 路径和无法打开终端直接进入相应文件夹的问题。

下载安装 Visual Studio Code 在这里 > Visual Studio Code | Code editing. Redefined.

问题

目前存在的一个问题是：VSCode 和 WSL 侧的工具链兼容性都很糟糕（除了 Node.js），都需要一定的配置才能丝滑工作。这也是一个当前微软 VSCode 各大语言插件组和 WSL 开发组都知道并在解决的问题（参考 VSCode Python 插件 Issue #67）。

由于 WSL 是一个 Runtime 环境，而 VSCode 只和 Windows 侧的组件进行沟通，因此当前一个比较好的解决方法是：在 Windows 侧手动创建一些脚本帮助

VSCode 和 WSL 侧安装的组件沟通。详见 Python 配置板块。

参考配置

为了方便参考，我使用的 Visual Studio Code 具体配置如下：

- 字体：
  - 附带有 Cursive 的字体叫做 Operator Mono，它

返回到第 13 页 第 49 页 第 50 页 本章还剩 2 页

制作出的电子书效果

工具使用

自制一本电子书需要 3 方面的准备，用到的工具和素材乍看

有点多，其实按部就班操作下来并不麻烦。

- **制作工具**：Pandoc，一个开源电子书制作、转换工具
- **书籍源文件**：有的网页提供素材下载，没有的话也可以复制全文。
- **书籍元数据**：作者、封面等信息，加上去才更像一本书，视角效果也更好。

下面我们一项一项来看。

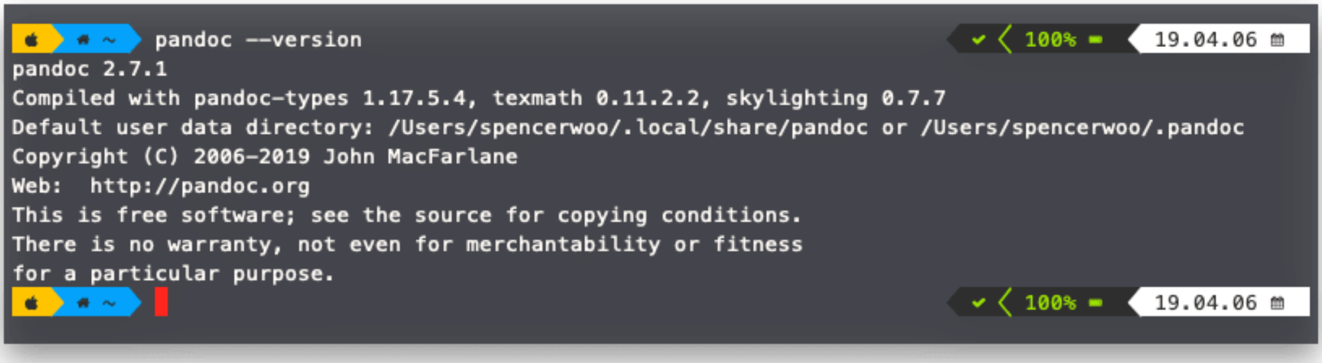
下载安装

Pandoc 是一个通用的文档转换工具，今天我们主要就利用 Pandoc 进行电子书制作。我们可以直接从 Pandoc 官网的下载页面 上面下载 Pandoc 的安装文件来安装这个工具。macOS 上使用 Homebrew 包管理的同学也可以直接使用下面这个命令安装：

```
brew install pandoc
```

Pandoc 这个工具是基于命令行的一个应用，因此我们需要在「终端」中使用这个工具。

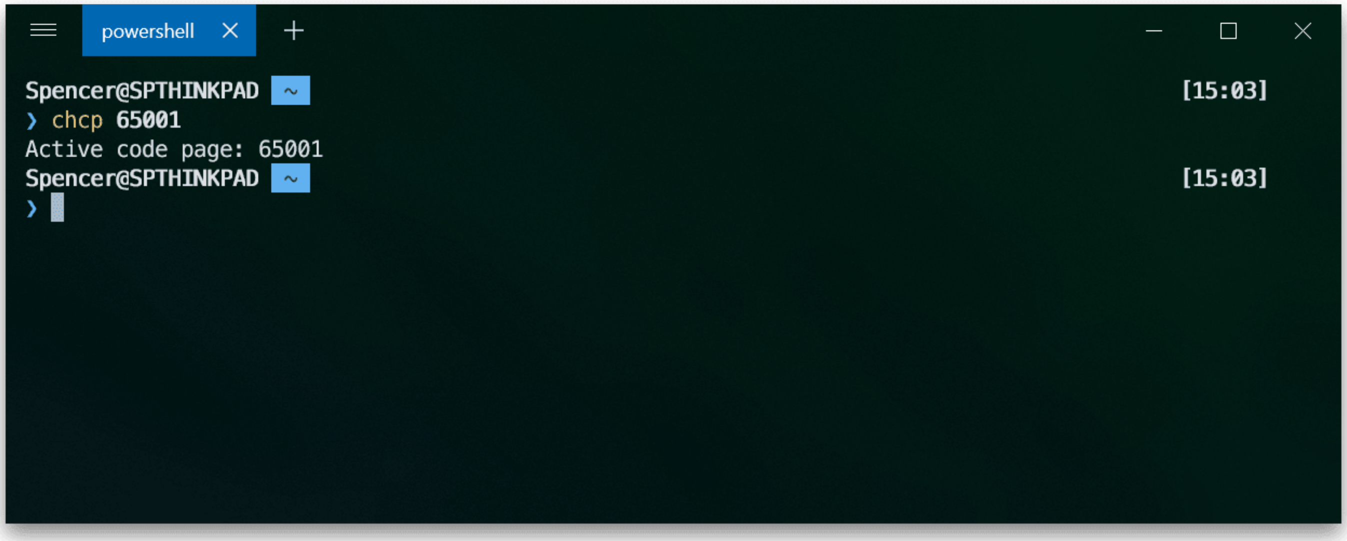
在 macOS 中，我们可以直接利用 Spotlight 搜索「终端」打开。我们可以直接输入 `pandoc --version` 来判断我们是否成功安装了这个工具，如果有类似下面的输出，那么我们安装就是成功的。



```
  pandoc --version                                        ✓ 〈 100% ▬   19.04.06
pandoc 2.7.1
Compiled with pandoc-types 1.17.5.4, texmath 0.11.2.2, skylighting 0.7.7
Default user data directory: /Users/spencerwoo/.local/share/pandoc or /Users/spencerwoo/.pandoc
Copyright (C) 2006-2019 John MacFarlane
Web:  http://pandoc.org
This is free software; see the source for copying conditions.
There is no warranty, not even for merchantability or fitness
for a particular purpose.
                                                          ✓ 〈 100% ▬   19.04.06
```

在 macOS 下安装 Pandoc

如果你使用的是 Windows，也可以直接使用 PowerShell 这个终端来运行 Pandoc。需要注意的是在 Windows 下，为了确保 Pandoc 输出正确，**我们需要首先执行命令 `chcp 65001`**。这样我们才能保证终端的输出环境为 UTF-8 的编码格式。



```
Spencer@SPTHINKPAD ~                                     [15:03]
❯ chcp 65001
Active code page: 65001
Spencer@SPTHINKPAD ~                                     [15:03]
❯
```

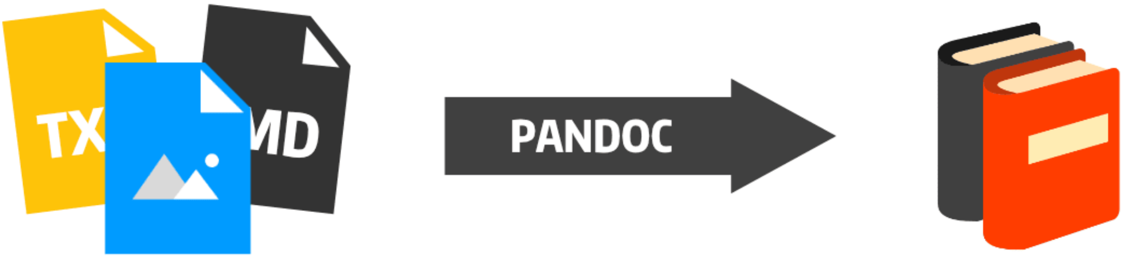
Windows 保证终端环境为 UTF-8 的编码格式

下面的制作过程在 macOS 和 Windows 上的操作步骤一致，我以 macOS 为例。

书籍源文件格式

在着手使用 Pandoc 制作电子书之前，容我再介绍一下一个合格的电子书源文件应该是怎样的格式和文件结构。

Markdown 作为一个简单、优秀的排版语言，是我们这次制作电子书的源语言。作为我派的忠实读者，大家一定对 Markdown 并不陌生了。Pandoc 实际上就是一个编译器，能够将 Markdown 格式的文件输出为电子书的形式，因此只要我们按照标准 Markdown 语法构建我们电子书源格式，就能辅助 Pandoc 为我们输出一本精美的电子书。



Pandoc 通用文档转换器

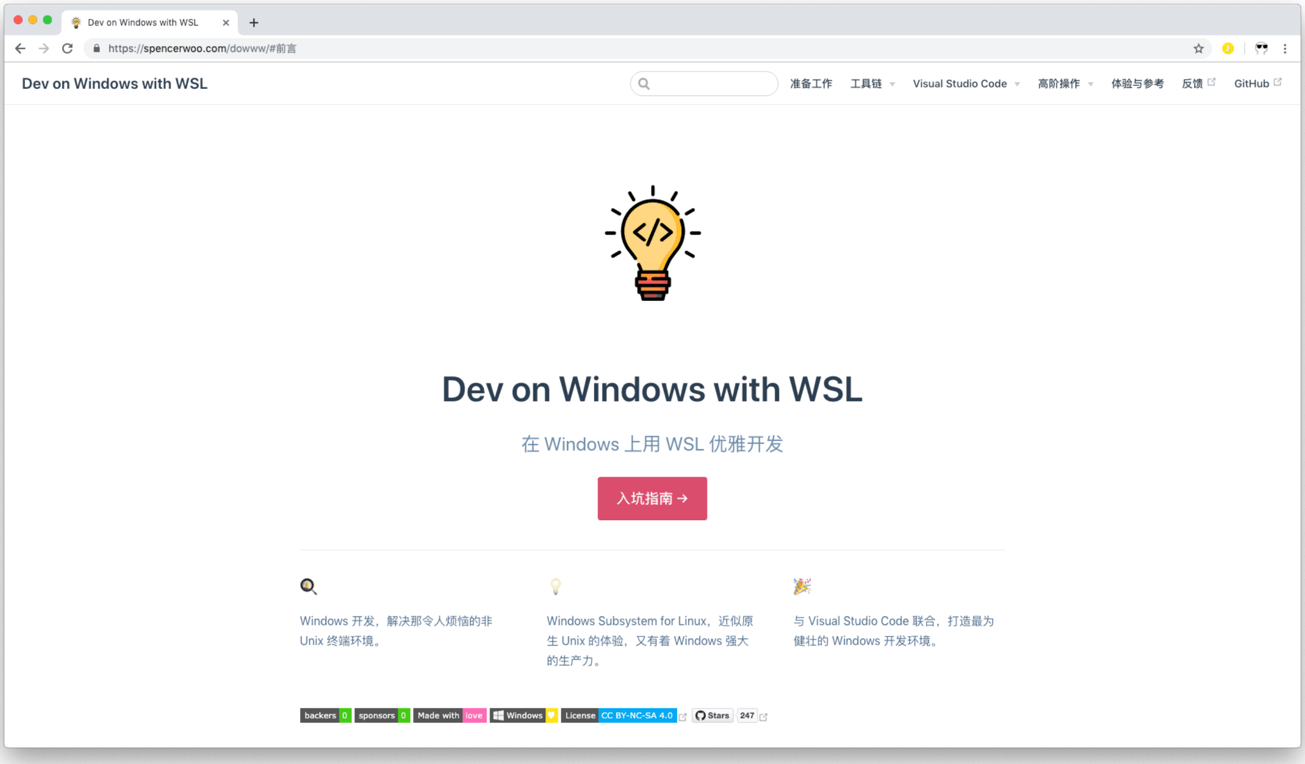
Pandoc 支持一个稍微不同的变种 Markdown (An extended and slightly revised Markdown syntax)，但是语法大体一致。不同的语法项目还请直接查看 Pandoc Markdown 官方文档，我在这里就不赘述了。

如果原本的电子书是 TXT 等纯文本格式，就需要我们自己手动为每一章节、每个标题等用 Markdown 语法重新制作，相对来说就比较繁琐。但是如果我们想要制作的电子书本身就是用 Markdown 书写的，那么制作就非常方便了，不太需要改动，一行命令就能解决问题。大多数开源的文档、开发手

册，甚至部分小说等，都是利用 Markdown 直接书写的。部署在静态网站上面的文章和参考指南等，其绝大多数源文件也是 Markdown。**因此很大程度上，我们并不需要太过担心，直接进行制作就可以了。**

制作过程

接下来，我以将我自己这个 [在 Windows 上面利用 WSL 高效开发]() 的参考指南转换为 ePub 形式的电子书为例子，为大家介绍如何输出一本电子书。


Dev on Windows with WSL
https://spencerwoo.com/dowww/#前言
Dev on Windows with WSL
准备工作
工具链
Visual Studio Code
高阶操作
体验与参考
反馈
GitHub
入坑指南 →
backers 0
sponsors 0
Made with love
Windows
License CC BY-NC-SA 4.0
Stars 247

Dev on Windows with WSL

在 Windows 上用 WSL 优雅开发

Windows 开发，解决那令人烦恼的非 Unix 终端环境。

Windows Subsystem for Linux，近似原生 Unix 的体验，又有着 Windows 强大的生产力。

与 Visual Studio Code 联合，打造最为健壮的 Windows 开发环境。

样例电子书在网页上的样子

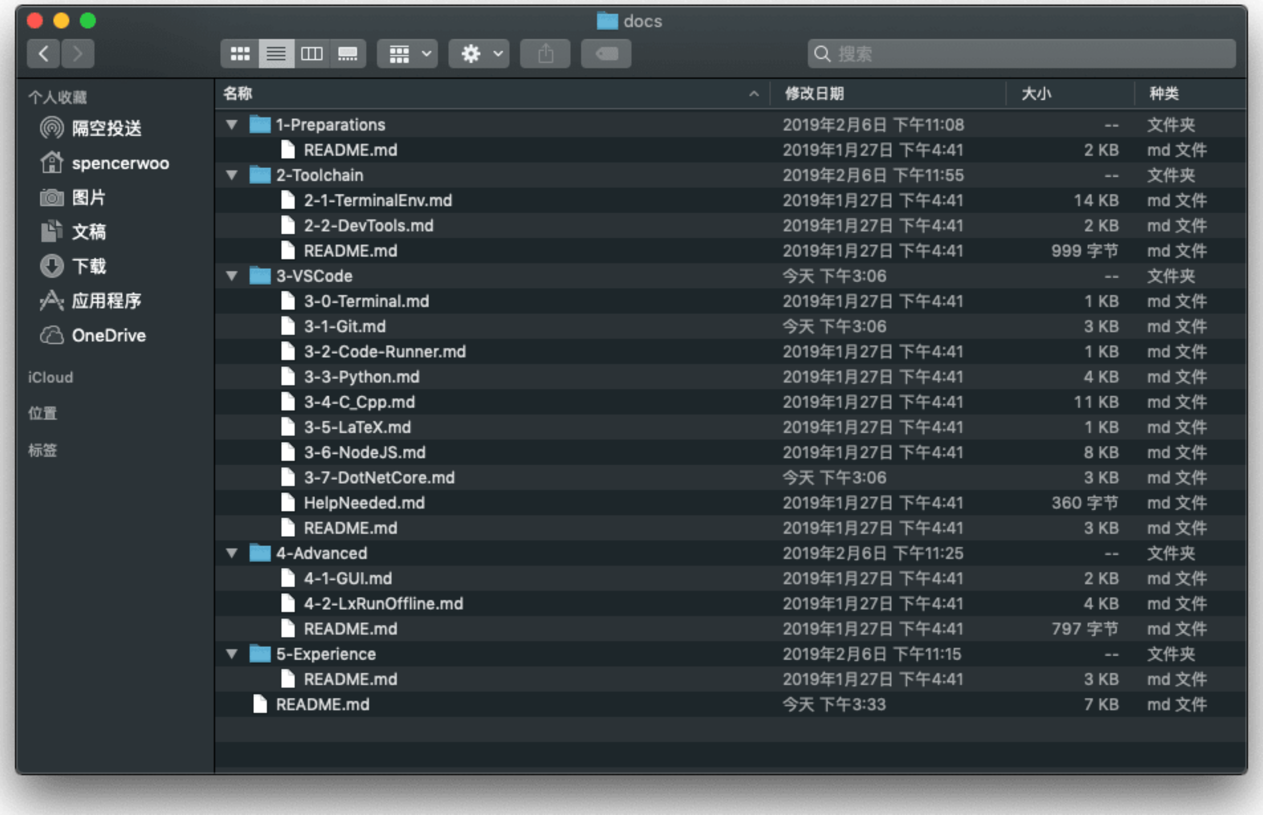
整理源文件

首先我们需要将「这本书」的**源文件**下载到本地。那么如何获取到我们希望转换的电子书的源文件呢？针对**开源书籍**和

普通网页，下面是两个常用的办法。

对于诸如我的这个参考指南之类的文档，其源文件大多都直接开源在了 GitHub，文档页面通常也会给出一个 GitHub 源链接。我「这本书」的源文件在 spencerwooo/dowww 这里，我们可以直接点击绿色按钮 Clone or download → Download ZIP 来下载整个项目的压缩包。使用 `git` 版本管理的同学当然也可以直接利用 `git clone https://github.com/spencerwooo/dowww.git` 这个命令将项目克隆至本地。

之后，我们打开（解压）刚刚下载好的文件夹，可以看到在 `docs` 文件夹下看到所有的源文件：



名称	修改日期	大小	种类
1-Preparations	2019年2月6日 下午11:08	--	文件夹
README.md	2019年1月27日 下午4:41	2 KB	md 文件
2-Toolchain	2019年2月6日 下午11:55	--	文件夹
2-1-TerminalEnv.md	2019年1月27日 下午4:41	14 KB	md 文件
2-2-DevTools.md	2019年1月27日 下午4:41	2 KB	md 文件
README.md	2019年1月27日 下午4:41	999 字节	md 文件
3-VSCode	今天 下午3:06	--	文件夹
3-0-Terminal.md	2019年1月27日 下午4:41	1 KB	md 文件
3-1-Git.md	今天 下午3:06	3 KB	md 文件
3-2-Code-Runner.md	2019年1月27日 下午4:41	1 KB	md 文件
3-3-Python.md	2019年1月27日 下午4:41	4 KB	md 文件
3-4-C_Cpp.md	2019年1月27日 下午4:41	11 KB	md 文件
3-5-LaTeX.md	2019年1月27日 下午4:41	1 KB	md 文件
3-6-NodeJS.md	2019年1月27日 下午4:41	8 KB	md 文件
3-7-DotNetCore.md	今天 下午3:06	3 KB	md 文件
HelpNeeded.md	2019年1月27日 下午4:41	360 字节	md 文件
README.md	2019年1月27日 下午4:41	3 KB	md 文件
4-Advanced	2019年2月6日 下午11:25	--	文件夹
4-1-GUI.md	2019年1月27日 下午4:41	2 KB	md 文件
4-2-LxRunOffline.md	2019年1月27日 下午4:41	4 KB	md 文件
README.md	2019年1月27日 下午4:41	797 字节	md 文件
5-Experience	2019年2月6日 下午11:15	--	文件夹
README.md	2019年1月27日 下午4:41	3 KB	md 文件
README.md	今天 下午3:33	7 KB	md 文件

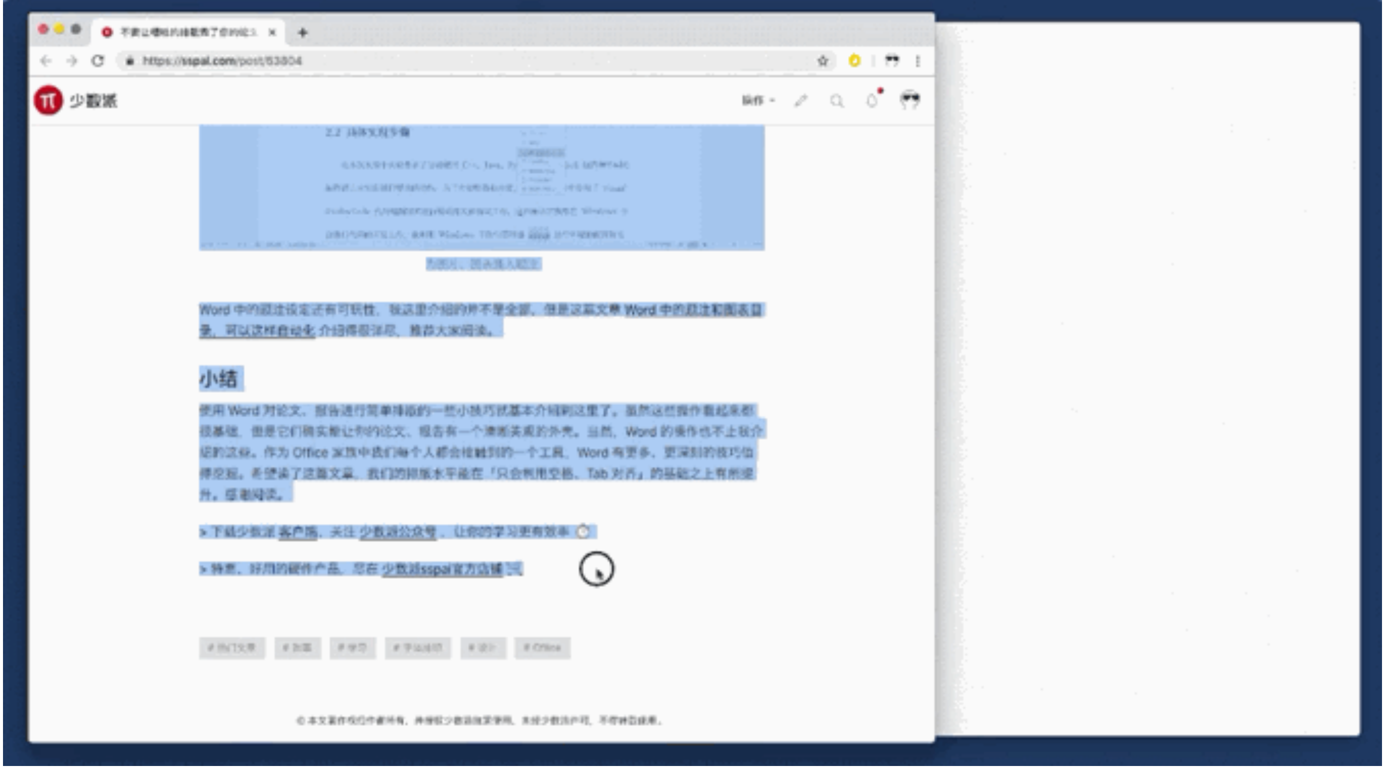
本次制作电子书的源文件

这些 Markdown 格式的源文件就是我们本次生成电子书的输入文件。

值得注意的是，我自己撰写这个文档的时候所有的图片都是用图床的形式存储在网上，因此 Markdown 内部的图片都是在线的外部链接，简称外链。无论是本地的相对路径引用格式的图片，还是图片外链，Pandoc 都能够正常的渲染，因此我们只要有图片的引用，就不用担心渲染出来电子书的图片样式。

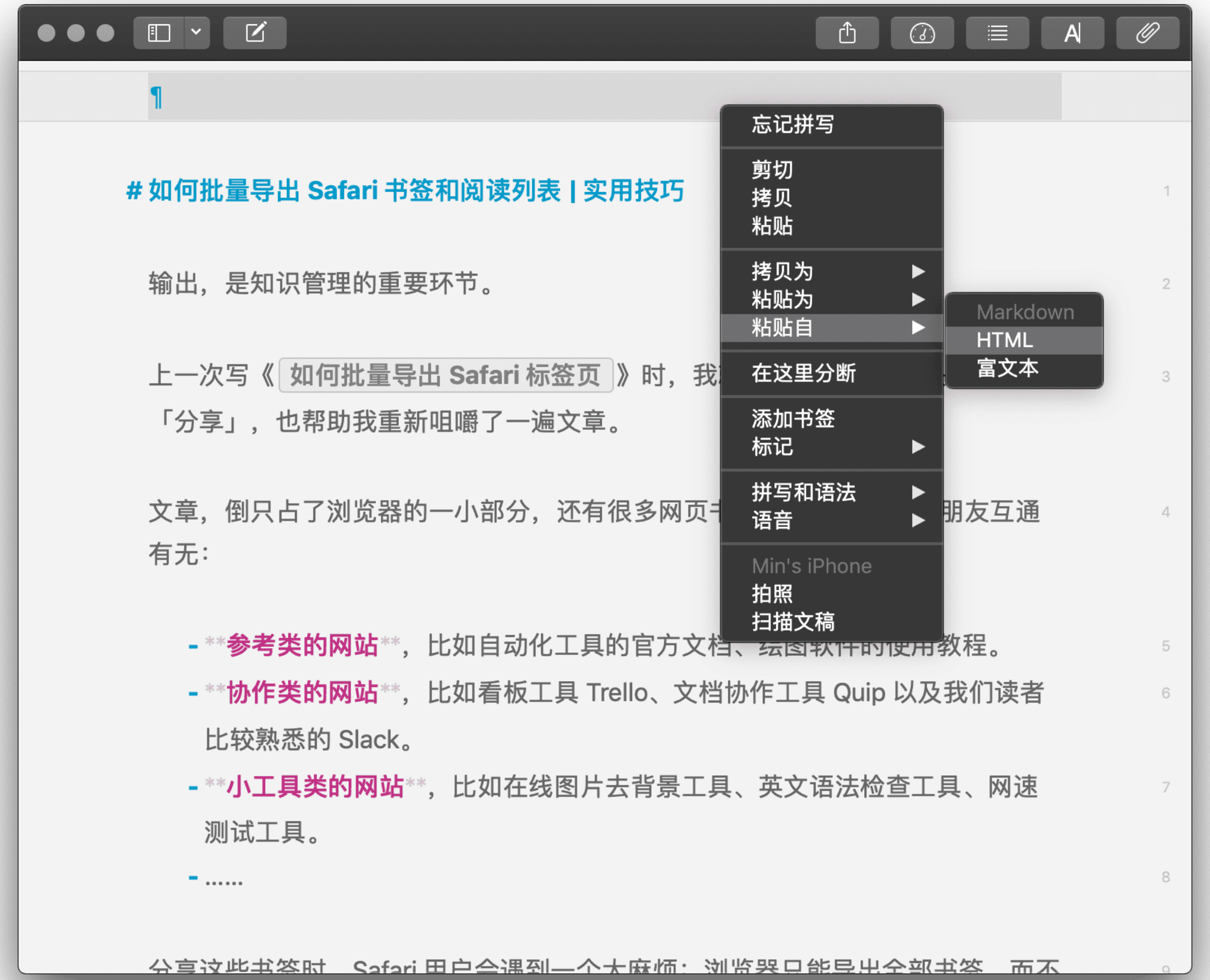
不过有时候，我们实在找不到书籍的源文件，或者书籍本身根本就不是开源的，则可以利用 Typora、Ulysses 等工具，将网页端的内容以 Markdown 的形式抓取到本地。

对于 Typora，我们可以直接复制网页端的全文，然后粘贴到 Typora 的新文档中，这样就可以以原格式的形式，直接将 Markdown 文件保存至本地了。Typora 不仅可以自动将网页上面复制的文本转换为 Markdown 的格式，就连图片的链接也能一并保存下来，极为方便。



用 Typora 连字带图抓取网页内容

Ulysses 同样也可以完成这样的操作。在网页上复制全文，新建一份 Ulysses 文档，右键选择**「粘贴自-HTML」**即可获得 Markdown 格式的文稿。网页自带的特殊元素（比如批注、提示）可能需要手动修剪一下。


¶
忘记拼写
剪切
拷贝
粘贴
拷贝为
粘贴为
粘贴自
在这里分断
添加书签
标记
拼写和语法
语音
Min's iPhone
拍照
扫描文稿
Markdown
HTML
富文本

如何批量导出 Safari 书签和阅读列表 | 实用技巧

输出，是知识管理的重要环节。

上一次写《如何批量导出 Safari 标签页》时，我
「分享」，也帮助我重新咀嚼了一遍文章。

文章，倒只占了浏览器的一小部分，还有很多网页
朋友互通有无：

- **参考类的网站**，比如自动化工具的官方文档、绘图软件的使用教程。
- **协作类的网站**，比如看板工具 Trello、文档协作工具 Quip 以及我们读者比较熟悉的 Slack。
- **小工具类的网站**，比如在线图片去背景工具、英文语法检查工具、网速测试工具。
- ……

分享这些书签时，Safari 用户会遇到一个大麻烦：浏览器只能导出全部书签，而不

把网页用 Markdown 格式拷贝进 Ulysses

Ulysses 倒是可以直接输出 epub 电子书，但是不能直接嵌入在线图片，在抓网页的场景中并不好用，最后仍然要借助 Pandoc。

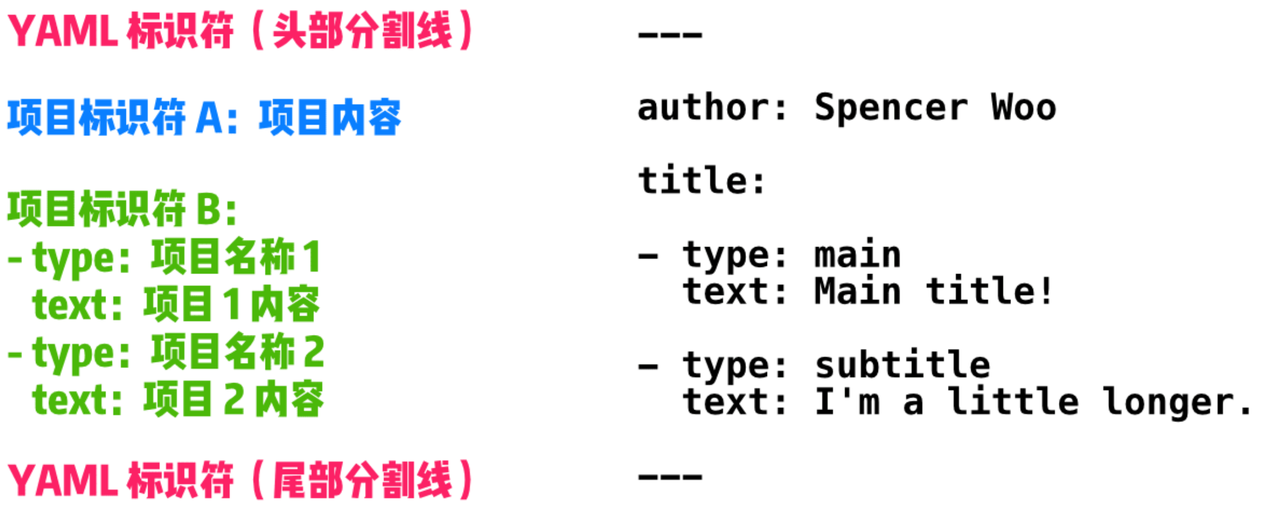
根据需要修改元数据

一本书不能没有封面、作者等信息，我们同样也需要通过「元数据」来为我们的电子书制定封面、作者、出版年份、版权等内容。

元数据，即 Metadata，在这里，是一段可以包含有这本电子书的标题（title）、创造者（creator）、出版商（publisher）、版权（rights）等等各种信息的文本。更多我们可以定义的项目可以直接参考：[Pandoc | ePub Metadata]。这里我们先定义封面图片、作者和版权这些信息，为大家示范。

首先，在刚刚的 `docs` 目录（或者你的电子书源文件根目录）创建一个名为 `title.txt` 的文件，并用「文本编辑」打开。

Pandoc 识别的元数据是 `YAML` 格式的文件，下面这张图是一个 `YAML` 文件的大致格式：


YAML 标识符（头部分割线）
项目标识符 A：项目内容
项目标识符 B：
- type：项目名称 1
text：项目 1 内容
- type：项目名称 2
text：项目 2 内容
YAML 标识符（尾部分割线）

```
---

author: Spencer Woo

title:

- type: main
  text: Main title!

- type: subtitle
  text: I'm a little longer.

---
```

YAML 的大致格式

可以看到，`YAML` 首先以 `---` 开始，也以 `---` 结束，这两行分割线中间就放置了我们要定义的内容。元数据的每一项都是 `项目名: 项目内容` 的格式，其中每个项目可以是一个单项目，也可以是一组数据（比如标题 `title` 可以有主标题 `main` 和副

标题 `subtitle` 等）的形式。

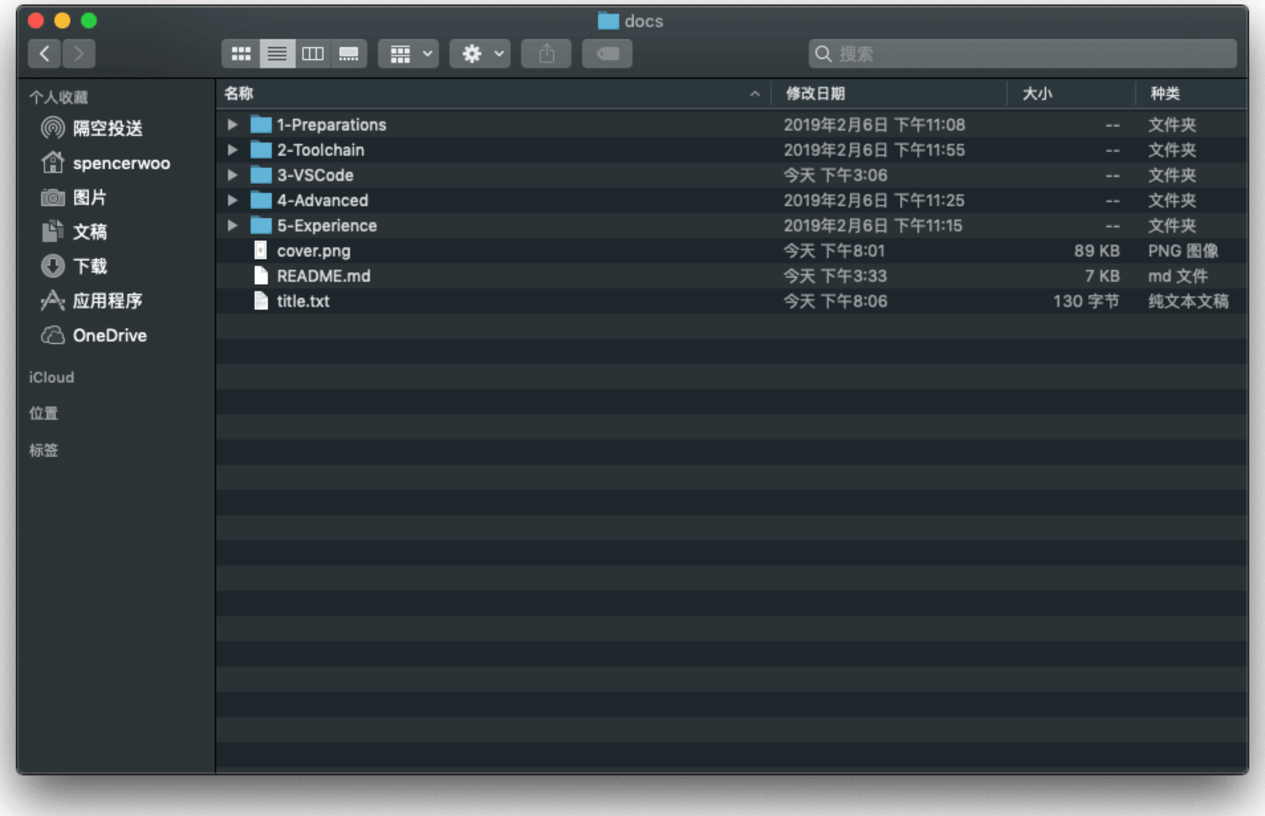
我们将标题 `title`、作者 `author`、日期 `date`、版权 `rights`、封面图片 `cover-image` 的信息如下整理：

```
---
title: 在 Windows 上用 WSL 优雅开发
author: Spencer Woo
date: 2019-04-06
rights: CC BY-NC-SA
cover-image: cover.png
---
```

我们将上面的内容存在刚刚创建 `title.txt` 中，就可以了。其中：

- `date` 日期这一项的格式是 `YYYY-MM-DD` 的格式
- `cover-image` 这一项封面图片的路径是相对路径，我们直接将图片 `cover.png` 放在电子书源文件根目录即可。

最终，我们的电子书源文件目录应该有这样的结构：


docs
搜索
个人收藏
隔空投送
spencerwoo
图片
文稿
下载
应用程序
OneDrive
iCloud
位置
标签

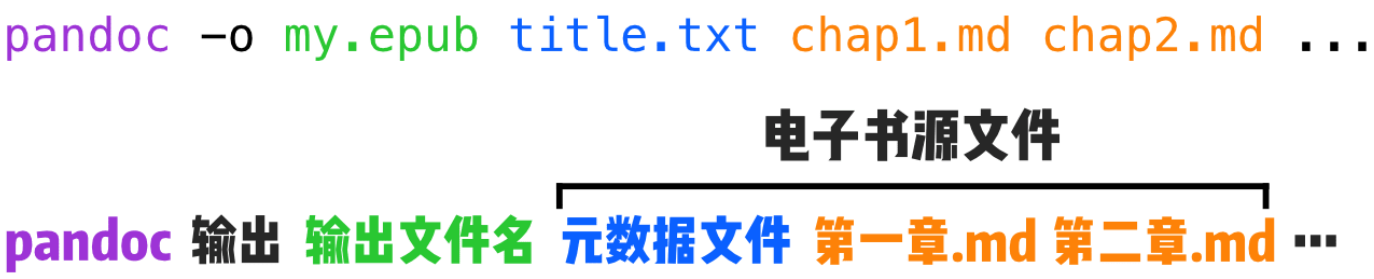
名称	修改日期	大小	种类
1-Preparations	2019年2月6日 下午11:08	--	文件夹
2-Toolchain	2019年2月6日 下午11:55	--	文件夹
3-VSCode	今天 下午3:06	--	文件夹
4-Advanced	2019年2月6日 下午11:25	--	文件夹
5-Experience	2019年2月6日 下午11:15	--	文件夹
cover.png	今天 下午8:01	89 KB	PNG 图像
README.md	今天 下午3:33	7 KB	md 文件
title.txt	今天 下午8:06	130 字节	纯文本文稿

电子书的源文件和元数据

制作输出电子书

这就是我们制作电子书最关键的一步了：将所有的 Markdown 格式的源文件、电子书的元数据和封面图片利用 Pandoc 制作一个 ePub 格式的电子书。

Pandoc 输出电子书的命令非常简单，只要我们上面几步准备好了电子书的源文件，这一步我们就需要一行命令即可直接制作一本电子书。


pandoc -o my.epub title.txt chap1.md chap2.md ...
电子书源文件
pandoc 输出 输出文件名 元数据文件 第一章.md 第二章.md ···

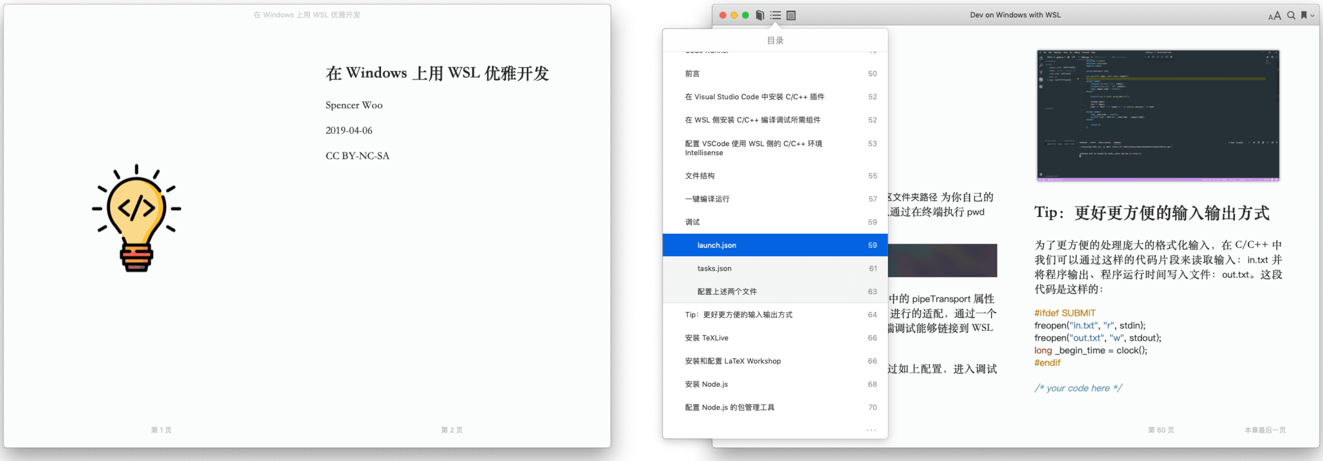
输出电子书的命令

利用 Pandoc 生成电子书的命令基本如上面这个结构。可以看到这个命令大致是 `pandoc -o <输出文件> <源文件>` 的结构。那么我们这里输出电子书的命令就很清晰了，我们定义输出文件名为 `dowww.epub`，将源文件的路径依次作为参数输入：

```
pandoc -o dowww.epub title.txt \
    1-Preparations/README.md \
    2-Toolchain/README.md \
    2-Toolchain/2-1-TerminalEnv.md \
    2-Toolchain/2-2-DevTools.md \
    3-VSCode/README.md \
    ...
    5-Experience/README.md
```

注：上面命令中的每一行末端的 并不是整个命令中必须的一部分，是为了让大家更清晰的看到命令，而不是一整行挤在一起的样子，加上的换行符号。在实际操作命令的时候，加上 也不会影响命令本身。

如果你想直接在生成的电子书中嵌入目录，在刚刚命令最后加上 `--toc` 就可以了。这样除了 ePub 阅读器提供的目录，电子书本身也内嵌了带有跳转链接的目录。



在 Windows 上用 WSL 优雅开发

在 Windows 上用 WSL 优雅开发
Spencer Woo
2019-04-06
CC BY-NC-SA

第 1 页 第 2 页

Dev on Windows with WSL

目录

前言	50
在 Visual Studio Code 中安装 C/C++ 插件	52
在 WSL 侧安装 C/C++ 编译调试所需组件	52
配置 VSCode 使用 WSL 侧的 C/C++ 环境 Intellisense	53
文件结构	55
一键编译运行	57
调试	59
launch.json	59
tasks.json	61
配置上述两个文件	63
Tip：更好更方便的输入输出方式	64
安装 TeXLive	66
安装和配置 LaTeX Workshop	66
安装 Node.js	68
配置 Node.js 的包管理工具	70

区文件夹路径 为你自己的
通过在终端执行 pwd

中的 pipeTransport 属性
进行的适配，通过一个
端调试能够链接到 WSL

过如上配置，进入调试

Tip：更好更方便的输入输出方式

为了更方便的处理庞大的格式化输入，在 C/C++ 中我们可以通过这样的代码片段来读取输入：in.txt 并将程序输出、程序运行时间写入文件：out.txt。这段代码是这样的：

```
#ifdef SUBMIT
freopen("in.txt", "r", stdin);
freopen("out.txt", "w", stdout);
long _begin_time = clock();
#endif

/* your code here */
```

第 60 页 本章最后一页

电子书成品

最后，我们就得到了排版整洁、图片清晰并且封面目录一应俱全的电子书了。🎉🎉🎉

进阶操作

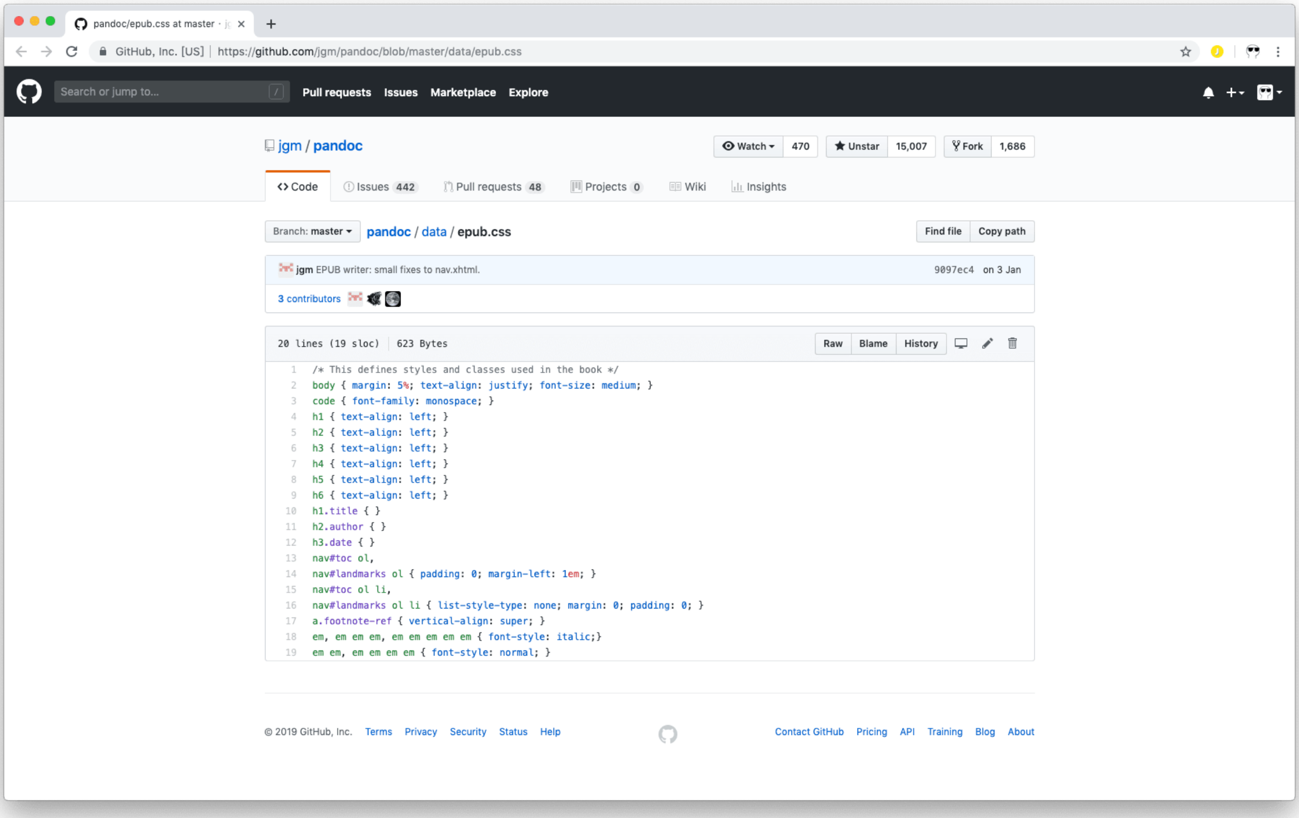
通过组合原始素材和元数据，我们得到的电子书已经比网页的可读性好了不少。不过 Pandoc 默认导出的电子书有点「素」，格式也仅限 ePub，对于样式和格式有更高需求的读者，可以在这一章中找到相应方法。

自定义主题

ePub 在本质上是一个压缩包，如果我们直接解压缩一个 ePub 文件，我们会得到 XHTML 格式的章节文件。在 Pandoc 生成电子书的过程中，最后渲染出来的电子书实际

上是利用 HTML 提供内容、框架，CSS 提供样式的格式生成的。因此，我们当然可以通过自定义 CSS 来定义生成的电子书样式。

Pandoc 默认的 CSS 是一个极简的样式，我们可以直接在 epub.css 这里看到整个 CSS 样式文件。

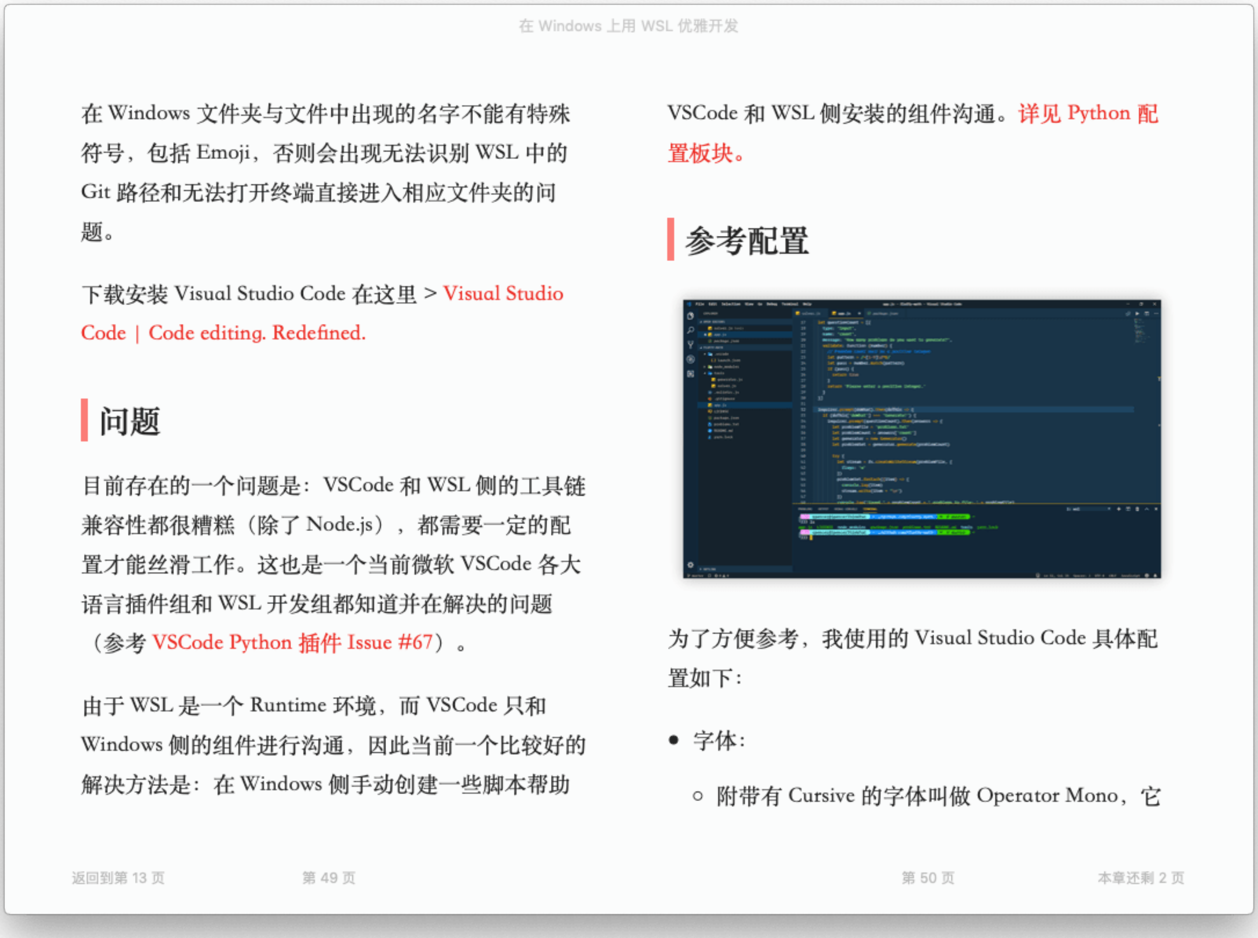


pandoc/epub.css at master · j
GitHub, Inc. [US] | https://github.com/jgm/pandoc/blob/master/data/epub.css
Search or jump to...
Pull requests
Issues
Marketplace
Explore
jgm / pandoc
Watch 470
Unstar 15,007
Fork 1,686
Code
Issues 442
Pull requests 48
Projects 0
Wiki
Insights
Branch: master
pandoc / data / epub.css
Find file
Copy path
jgm EPUB writer: small fixes to nav.xhtml.
9097ec4 on 3 Jan
3 contributors
20 lines (19 sloc) 623 Bytes
Raw
Blame
History
© 2019 GitHub, Inc. Terms Privacy Security Status Help
Contact GitHub Pricing API Training Blog About

```css
/* This defines styles and classes used in the book */
body { margin: 5%; text-align: justify; font-size: medium; }
code { font-family: monospace; }
h1 { text-align: left; }
h2 { text-align: left; }
h3 { text-align: left; }
h4 { text-align: left; }
h5 { text-align: left; }
h6 { text-align: left; }
h1.title { }
h2.author { }
h3.date { }
nav#toc ol,
nav#landmarks ol { padding: 0; margin-left: 1em; }
nav#toc ol li,
nav#landmarks ol li { list-style-type: none; margin: 0; padding: 0; }
a.footnote-ref { vertical-align: super; }
em, em em em, em em em em em { font-style: italic;}
em em, em em em em { font-style: normal; }
```

epub.css

在编译电子书时，我们可以通过在命令行参数上添加 `--css <样式文件>.css` 这个选项添加自定义样式。比如，我使用少数派的 CSS 主题：少数派经典 CSS 主题，利用 Pandoc 进行编译整理，最后就能得到这样有「一抹红」效果的电子书。



在 Windows 上用 WSL 优雅开发

在 Windows 文件夹与文件中出现的名字不能有特殊符号，包括 Emoji，否则会出现无法识别 WSL 中的 Git 路径和无法打开终端直接进入相应文件夹的问题。

下载安装 Visual Studio Code 在这里 > Visual Studio Code | Code editing. Redefined.

问题

目前存在的一个问题是：VSCode 和 WSL 侧的工具链兼容性都很糟糕（除了 Node.js），都需要一定的配置才能丝滑工作。这也是一个当前微软 VSCode 各大语言插件组和 WSL 开发组都知道并在解决的问题（参考 VSCode Python 插件 Issue #67）。

由于 WSL 是一个 Runtime 环境，而 VSCode 只和 Windows 侧的组件进行沟通，因此当前一个比较好的解决方法是：在 Windows 侧手动创建一些脚本帮助 VSCode 和 WSL 侧安装的组件沟通。详见 Python 配置板块。

参考配置

为了方便参考，我使用的 Visual Studio Code 具体配置如下：

- 字体：
  - 附带有 Cursive 的字体叫做 Operator Mono，它

返回到第 13 页　第 49 页　第 50 页　本章还剩 2 页

sspai.css 效果

如果你找不到合适的 CSS 主题，推荐参考 Minja 整理的：[简单又好看，你的 Markdown 文稿也能加上个性化主题]

生成 MOBI 格式的书籍

正如前文提到的，MOBI 格式主要是用在 Kindle 等电子书阅读器上面的格式。我们可以通过 Amazon 为我们提供的命令行工具 KindleGen 将 ePub 格式的电子书转换成 MOBI 格式，并发送到 Kindle 上面查看。

我们可以在 KindleGen 介绍页面 上面直接下载 KindleGen 这个工具。使用 Homebrew 的同学也可以通过下面这个命令下

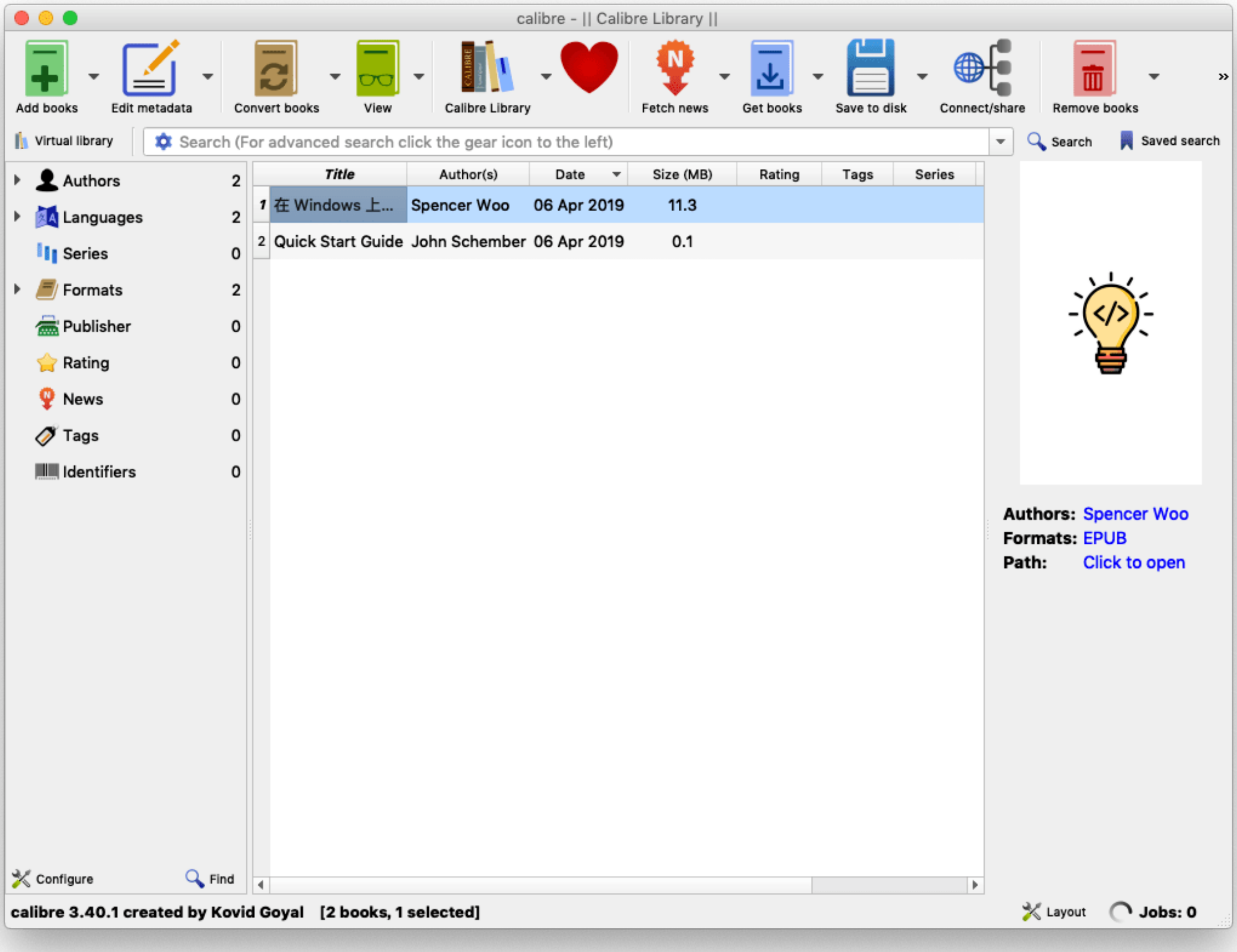
载 KindleGen：

```
brew cask install kindlegen
```

之后，简单的一行命令就可以将 ePub 格式的电子书转换为 MOBI 格式的 Kindle 电子书：

```
kindlegen <原电子书名>.epub -o <输出电子书名>.mobi
```

不过，KindleGen 在大陆访问起来不太顺利，命令行工具的使用也不太人性化，因此如果你不想使用 KindleGen 进行格式转换，我们也可以**利用 Calibre 进行电子书的管理生成。**


calibre - || Calibre Library ||
Add books
Edit metadata
Convert books
View
Calibre Library
Fetch news
Get books
Save to disk
Connect/share
Remove books
Virtual library
Search (For advanced search click the gear icon to the left)
Search
Saved search
Configure
Find
Layout
Jobs: 0

Authors	2
Languages	2
Series	0
Formats	2
Publisher	0
Rating	0
News	0
Tags	0
Identifiers	0

	Title	Author(s)	Date	Size (MB)	Rating	Tags	Series
1	在 Windows 上...	Spencer Woo	06 Apr 2019	11.3			
2	Quick Start Guide	John Schember	06 Apr 2019	0.1			

Authors: Spencer Woo
Formats: EPUB
Path: Click to open

calibre 3.40.1 created by Kovid Goyal [2 books, 1 selected]

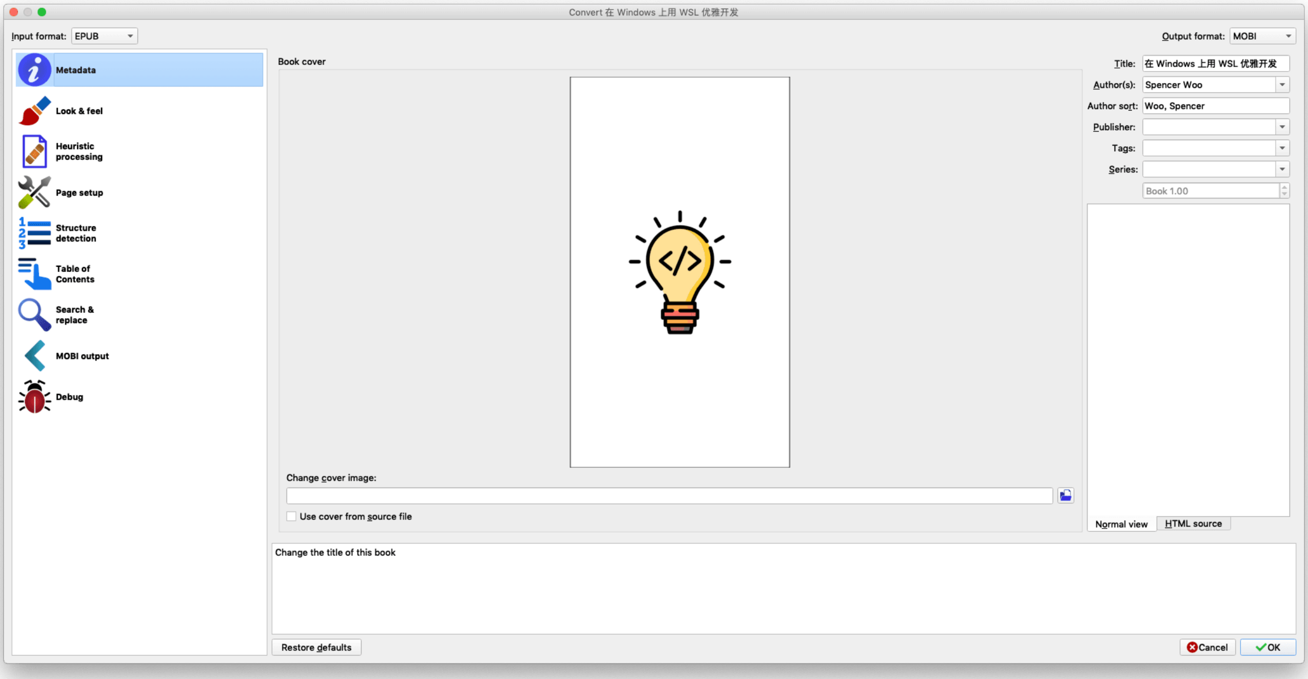
用 Calibre 制作电子书

Calibre 是「一站式电子书管理器」，我们不仅可以利用 Calibre 进行格式转换，还可以直接用 Calibre 加载、下载、保存各种电子书。我们可以直接在 Calibre 的官网 下载安装 Calibre。也可以利用 Homebrew 直接下载：

```
brew cask install calibre
```

之后，在主页面我们先导入刚刚生成的 ePub 格式的电子书，选择这本书。之后，选择「转换书籍」，将输入格式设置为 ePub，输出格式设置为 MOBI，我们就直接得到了一本

MOBI 格式的电子书。



Convert 在 Windows 上用 WSL 优雅开发

Input format: EPUB

Output format: MOBI

Title: 在 Windows 上用 WSL 优雅开发

Author(s): Spencer Woo

Author sort: Woo, Spencer

Publisher:

Tags:

Series:

Book 1.00

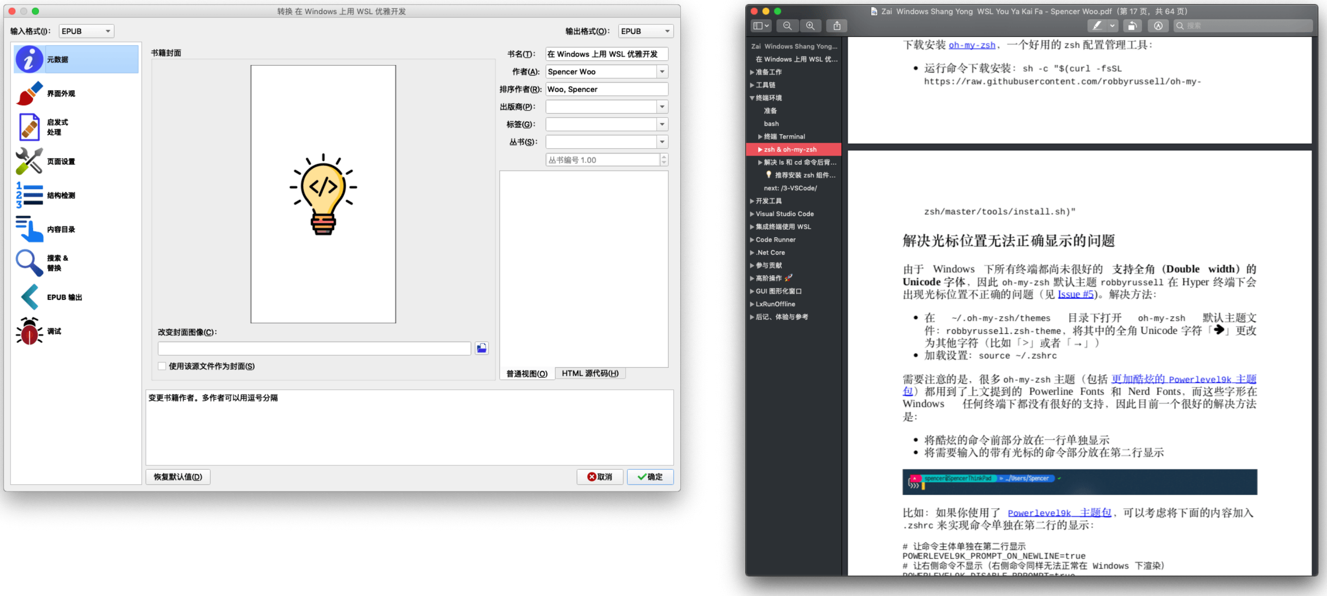
Change the title of this book

Metadata
Look & feel
Heuristic processing
Page setup
Structure detection
Table of Contents
Search & replace
MOBI output
Debug
Book cover
Change cover image:
Use cover from source file
Normal view
HTML source
Restore defaults
Cancel
OK

通过 Calibre 制作 MOBI 电子书

生成 PDF 格式的文档

利用 Pandoc 我们也可以输出 PDF 格式的文档，但是 Pandoc 的 PDF 生成器依赖于 LaTeX 编译器才能正常渲染，大多数同学如果不写 LaTeX 文档，并不会折腾 LaTeX，因此这一途径相对比较麻烦。不过，幸好 Calibre 支持有生成 PDF 文档的选项，因此我们可以直接利用 Calibre 将 ePub 格式的电子书转换为 PDF 文档。


转换 在 Windows 上用 WSL 优雅开发
元数据
界面外观
启发式处理
页面设置
结构检测
内容目录
搜索 & 替换
EPUB 输出
调试
书籍封面
普通视图(O)
HTML 源代码(H)
恢复默认值(D)
取消
确定
Zai Windows Shang Yong WSL You Ya Kai Fa - Spencer Woo.pdf（第 17 页，共 64 页）
搜索
Zai Windows Shang Yong...
在 Windows 上用 WSL 优...
准备工作
工具链
终端环境
准备
bash
终端 Terminal
zsh & oh-my-zsh
解决 ls 和 cd 命令后背...
推荐安装 zsh 组件...
next: /3-VSCode/
开发工具
Visual Studio Code
集成终端使用 WSL
Code Runner
.Net Core
参与贡献
高阶操作 🚀
GUI 图形化窗口
LxRunOffline
后记、体验与参考

输入格式(I)：EPUB
输出格式(O)：EPUB
书名(T)：在 Windows 上用 WSL 优雅开发
作者(A)：Spencer Woo
排序作者(R)：Woo, Spencer
出版商(P)：
标签(G)：
丛书(S)：
丛书编号 1.00
改变封面图像(C)：
☐ 使用该源文件作为封面(S)

变更书籍作者。多作者可以用逗号分隔

下载安装 oh-my-zsh，一个好用的 zsh 配置管理工具：

- 运行命令下载安装：sh -c "$(curl -fsSL https://raw.githubusercontent.com/robbyrussell/oh-my-zsh/master/tools/install.sh)"

解决光标位置无法正确显示的问题

由于 Windows 下所有终端都尚未很好的 支持全角（**Double width**）的 **Unicode** 字体，因此 oh-my-zsh 默认主题 robbyrussell 在 Hyper 终端下会出现光标位置不正确的问题（见 Issue #5）。解决方法：

- 在 ~/.oh-my-zsh/themes 目录下打开 oh-my-zsh 默认主题文件：robbyrussell.zsh-theme，将其中的全角 Unicode 字符「➔」更改为其他字符（比如「>」或者「→」）
- 加载设置：source ~/.zshrc

需要注意的是，很多 oh-my-zsh 主题（包括 更加酷炫的 Powerlevel9k 主题包）都用到了上文提到的 Powerline Fonts 和 Nerd Fonts，而这些字形在 Windows 任何终端下都没有很好的支持，因此目前一个很好的解决方法是：

- 将酷炫的命令前部分放在一行单独显示
- 将需要输入的带有光标的命令部分放在第二行显示

```
spencer@SpencerThinkPad  ~/Users/Spencer
```

比如：如果你使用了 Powerlevel9k 主题包，可以考虑将下面的内容加入 .zshrc 来实现命令单独在第二行的显示：

```
# 让命令主体单独在第二行显示
POWERLEVEL9K_PROMPT_ON_NEWLINE=true
# 让右侧命令不显示（右侧命令同样无法正常在 Windows 下渲染）
```

利用 Calibre 将 ePub 格式的电子书转换为 PDF

如果 ePub 文件本身有目录，通过 Calibre 转换出的 **PDF 文件也会配备目录**，便于翻阅。从 MOBI 和 PDF 文档的生成也可以看出，ePub 格式的确具有很好的可塑性。

小结

借助于 Pandoc 这个优秀的「万能文档转换器」，我们能够方便的自己制作内容全面、格式精美、图片丰富的电子书，再借助于 Calibre 这个可视化的电子书管理应用，我们能让我们制作的电子书在任何平台上查看。

在这个越来越「无纸化」的时代，能让自己撰写的书籍在各种设备上获得同样的阅览效果，也是一种享受。希望这篇文章能够让你制作、阅读电子书的体验有所进步。感谢阅读。